MATERIAL INTERFACE RECONSTRUCTION

Kathleen S. Bonnell[1]

Mark A. Duchaineau[2]
Daniel R. Schikore[3]

Bernd Hamann[4]
Kenneth I. Joy[5]

Abstract

This paper presents a new algorithm for material interface reconstruction for data sets where fractional material information is given as a percentage for each element of the underlying mesh. The reconstruction problem is transformed to a problem that analyzes a dual data set, where each vertex in the dual mesh has an associated barycentric coordinate tuple that represents the fraction of each material present. After constructing the dual mesh from the original mesh, material boundaries are constructed by mapping a simplex into barycentric space, calculating the intersections with Voronoi cells that represent the regions where one material dominates. These intersections are mapped back to the original space and triangulated to form a boundary surface approximation. This algorithm can be applied to any grid structure and can treat any number of materials per element. It is a generalization of previous work, extending the reconstruction to three-dimensional grids, generating continuous surfaces for the boundary representation, and allowing for any number of materials to be present. Error analysis shows that the algorithm preserves volume fractions within an error range of 0.5% per material.

Keywords: Eulerian flow, material boundaries, finite elements, barycentric coordinates, volume fraction, isosurface extraction.

[1]P.O. Box 808, L-312, Lawrence Livermore National Laboratory, Livermore, CA 94551, USA; e-mail: `ksbonnell@ucdavis.edu`

[2]Center for Advanced Scientific Computing (CASC), Lawrence Livermore National Laboratory, Livermore, CA 94551, USA; e-mail: `duchaine@llnl.gov`

[3]Computational Engineering International, Morrisville, NC 27560, USA; e-mail: `schikore@ceintl.com`

[4]Corresponding Author, Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California, Davis, CA 95616-8562, USA; e-mail: `joy@cs.ucdavis.edu`

[5]Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California, Davis, CA 95616-8562, USA; e-mail: `hamann@cs.ucdavis.edu`

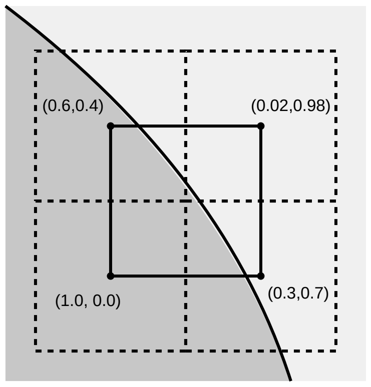


FIGURE 1: Original grid and dual grid. The original grid (dashed lines) is replaced by a dual grid (solid lines), obtained by connecting the centers of the original elements. Barycentric coordinates are associated with each vertex of the dual grid. The barycentric coordinates represent the fractions of each material associated with the original grid cells.

1. INTRODUCTION

In many applications it is necessary to reconstruct or track the boundary surfaces (or "interfaces") between multiple materials that commonly result from finite-element simulations. Multi-fluid Eulerian hydrodynamics calculations require geometric approximations of fluid interfaces to form the equations of motion to advance interfaces over time. Typically, the grid cells (finite elements) contain fractional information for each of the materials. Each cell C of a grid $\mathcal{S}$ has an associated tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$ that represents the portions of each of m materials in the cell, *i.e.*, α_i represents the fractional part of material i. It is assumed that $\alpha_1 + \alpha_2 + \cdots + \alpha_m = 1$ and $\alpha_i \geq 0$. Given the fractions for each cell, we wish to find a crack-free piecewise two-manifold surface approximating the boundary surfaces between the various materials.

To solve this problem, we consider the dual mesh constructed from the original mesh, as shown in Figure 1. In the dual grid, each cell is represented by a point (typically the center of the cell), and each point has an associated tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$, where m is the number of materials present in the data set. Thus, the boundary surface reconstruction problem reduces to constructing the material interfaces for a grid where each vertex has an associated barycentric coordinate tuple representing the fractional parts of each material. This "barycentric coordinate field" is used to approximate the material boundary surfaces.

Important applications of this problem occur for all grid types, *e.g.*, rectilinear, curvilinear, or unstructured grids. Therefore, we developed a solution strategy that is tailored to tetrahedral grids, as all other types of three-dimensional grid structures can be converted to this form, see Nielson [1]. In the case of rectilinear, curvilinear, or even hybrid polyhedral meshes, a given grid is pre-processed by subdividing each polyhedral cell into tetrahedra and applying this algorithm to the resulting tetrahedral grid.

Given a data set containing m materials, we process each tetrahedral cell of the grid and map our tetrahedral elements into an m simplex representing m-dimensional barycentric space. Next, we calculate intersections with the edges of *Voronoi cells* [2] in the m-simplex. These Voronoi cells represent regions, where one material "dominates" the other materials locally. We map these intersections back to the original space and triangulate the resulting points to obtain a boundary approximation.

We limit our discussion to three-dimensional grids. However, the techniques extend directly to multi-dimensional grids. In these cases, we convert the grid to a set of n-dimensional simplices, mapping these simplices into barycentric space. The intersections with a corresponding set of Voronoi cells can be calculated directly and mapped back to the original space. Triangulating the results is straightforward.

Section 2 describes previous work dealing with reconstruction of material boundary surfaces. Section 3 describes the two-material case, which can be viewed as a simple extension of a isosurface extraction technique. [3, 4, 5]. Section 4 describes the three-material case. Here, material boundaries are calculated in barycentric space (a triangle) and mapped back to the original data set. The general m-material case is described in Section 5. In this case, intersections are calculated in a barycentric m-simplex and mapped back to the tetrahedra in the data set. Implementation details are described in Section 6. Section 7 presents results for various data sets, and Section 8 provides error analysis.

2. RELATED WORK

Most research in material interface reconstruction has been conducted in computational fluid dynamics (CFD) and hydrodynamics, where researchers are concerned with the movement of material boundaries during a simulation. The *Simple Line Interface Calculation* (SLIC) algorithm by Noh and Woodward [6] is one of the earliest algorithms, describing a method for geometric approximation of fluid interfaces. Their algorithm is used in conjunction with hydrodynamics simulations to track the advection of fluids. It produces an interface consisting of line segments, constructed parallel or perpendicular to a coordinate axis. Multi-fluid cells can be handled by grouping fluids together, calculating the interface between the groups, subdividing the groups, and iterating this process. Since this algorithm only uses line segments that are parallel to the coordinate axes, the resulting interfaces are generally discontinuous.

In determining the direction of the line segment, cells to the left and right (in the appropriate coordinate direction) of the current cell are considered, and classified according to the fluid index. The fluid index indicates the presence (1) or absence (0) of a material. Mixed-fluid cells have multiple fluid indices, one per material. Fluids with the same fluid index are grouped together so that only two types may be treated at one time.

Consider a two-fluid 2D cell consisting of materials A (30%) and B (70%), and an x-direction pass of the algorithm, If the left neighbor contains only material A and the right neighbor contains only material B, the algorithm will generate an interface in the mixed fluid cell approximated by a vertical line dividing the cell into two regions, 30% A on the left and 70% B on the right. If that same cell has both left and right neighbors consisting entirely of material A, then the interface in the mixed-material cell would consist of two vertical lines dividing the

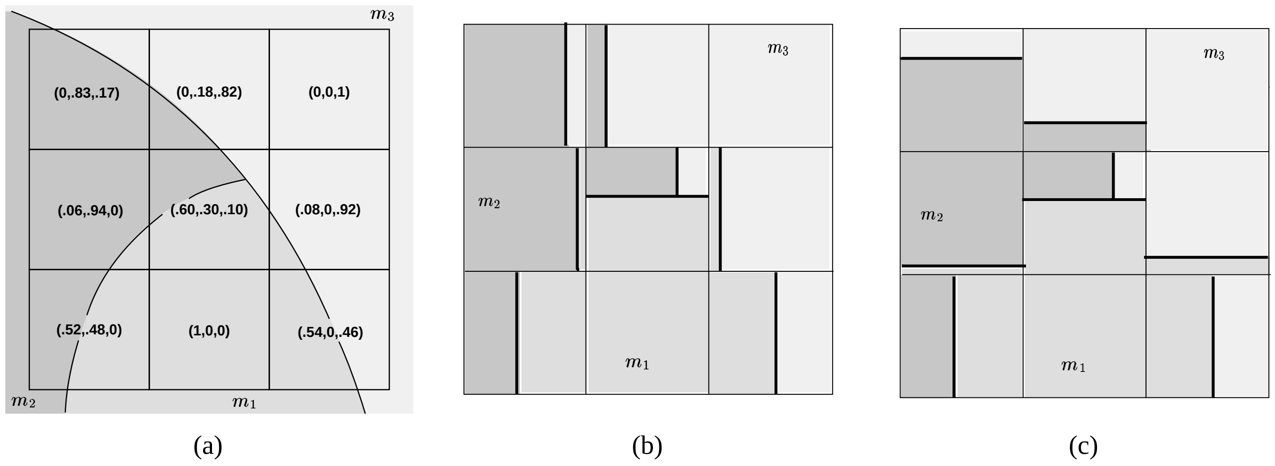


FIGURE 2: Example data set (a). The specific volume fractions are listed for each cell. The approximations generated by the SLIC algorithm are shown in (b) and (c), using an x-pass and y-pass, respectively.

cell into three parts: 15% material A on the left, 70% material B in the center, and 15% material A on the right.

Consider a three-fluid cell containing 20% A, 45% B, and 35% C, again with an x-direction pass. If the left neighbor consists entirely of material A and the right neighbor consists entirely of material B, then the interface would be represented by two vertical lines, dividing the cell into three zones, with material A on the left, C in the middle, and B on the right. If the left neighbor contains both A and B and the right neighbor contains A and C, then a horizontal line segment is used to first construct the zone with material A, then the remaining portion of the cell is divided by a vertical line with B on the left and C on the right. These simple rules can be used to generate material interfaces for two-dimensional rectilinear grids. Figure 2 shows the approximation generated by the SLIC algorithm, with both an x-coordinate and y-coordinate pass. Although the interface is discontinuous, the volume fractions are preserved for each cell.

The algorithm of Youngs [7] also operates on two-dimensional grids and uses line segments to approximate interfaces. In this algorithm, the line segments are not necessarily perpendicular or parallel to a coordinate axis. Instead, the neighbor cells of a cell C are used to determine the slope of a line segment approximating an interface in C. The exact location of the line segment is adjusted to preserve volume fractions. Multiple materials are treated by grouping materials and determining interfaces on a two-material basis. Again, the resulting interfaces are generally discontinuous.

Since this algorithm treats only two materials (or groups of materials) at a time, one of the materials is used to determine the slope of the interface line. This is done by using neighboring fractions of this material, and the Pythagorean theorem. Figure 3 demonstrates the neighbors of a cell containing materials A and B. The cell is treated as a unit-square for this calculation. The slope is defined as $\sqrt{(\delta_A - \gamma_A)^2 + (\beta_A - \alpha_A)^2}$, where α_A, β_A,

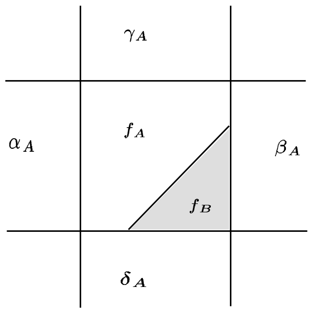


FIGURE 3: Two-material cell with material fractions f_A and f_B. Neighbor cells have material fractions α_A, β_A, γ_A, and δ_A for material A. The slope of the line is determined by the values of α_A, β_A, γ_A, and δ_A. The position of the line is defined by f_A and f_B

γ_A, and δ_A are the fractions of material A present in the neighbor cells[1] Once the slope is determined, the line segment is positioned in the cell such that the volume fractions are preserved. For more than two materials, the user determines in which order interfaces are calculated. Different interfaces will result depending on the chosen ordering/grouping. Figure 4 demonstrates the results of applying Youngs' algorithm.

The algorithm of Gueyffier [8] is similar to that of Youngs in that it requires an estimate of the normal vector to the interface in order to reconstruct the interface. Geuyffier's method utilizes finite-differencing or least-squares methods to determine this normal, depending upon the order of accuracy (first- or second-order) desired. In 2D, a line segment representing the boundary surface is constructed perpendicular to the interface normal. The line segment is positioned in the cell such that it divides the cell into appropriately proportioned areas. In the 3D case, a cutting plane is computed whose normal is the interface normal. Again, the cutting plane is positioned in the cell so that volume fractions are preserved. It is unclear how this algorithm would handle multiple materials.

Pilliod and Puckett [9] compare various volume-of-fluid interface reconstruction algorithms, including SLIC, noting differences in the surfaces reconstructed and demonstrating first-order or second-order accuracy. Their goal is to develop an algorithm that accurately reproduces a linear material interface, allowing discontinuous interfaces if the material boundary is not linear.

Nielson and Franke [10] have presented a method for calculating a separating surface in an unstructured grid where each vertex of the grid is associated with one of several possible classes. Their method generalizes the marching-cubes (or marching-tetrahedra) algorithm, but instead of using a strict binary classification of vertices, it allows any number of classes. Edges in tetrahedral grids whose endpoints have different classifications are intersected by the separating surface. Similarly, the faces of a tetrahedron whose three vertices are classified differently, are assumed to be intersected by the surface in the middle of the face. When all four vertices of a tetrahedron have different classifications, the boundary surface intersects in the interior of the tetrahedron. The

[1]We may use neighboring "corner" cells to determine the slope if this equation fails. Also, one may have to use the "negative" square root to determine the correct slope.

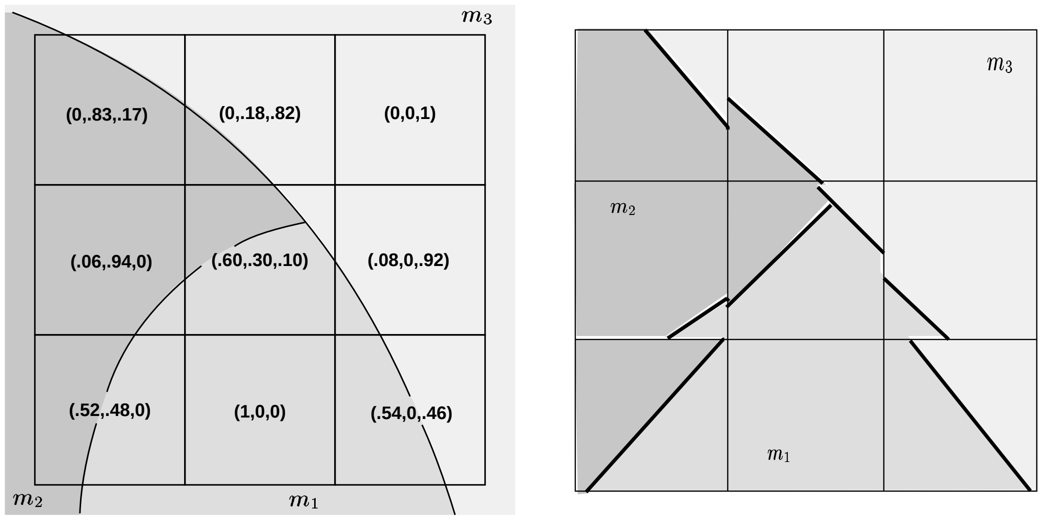


FIGURE 4: Result of Youngs' algorithm applied to the data shown in (a). The material interface representation (b) is discontinuous.

resulting "mid-edge," "mid-face," and "mid-tetrahedron" intersections are triangulated to linearly approximate the surface.

Using the dual-grid representation, Figure 5 shows how Nielson and Franke's algorithm might be applied to the example data set. As this algorithm requires classification of vertices, the greater α_i value in the volume fraction tuple for each cell was chosen as the classifier. The dual mesh has also been triangulated so that the algorithm can be applied. Mid-edge intersections are made half-way between two endpoints that have different classifications. Mid-face intersections are assumed when all three vertices of the triangle have different classifications.

This paper is an expanion of our work discussed in [11]. Our algorithm generalizes the above schemes. It utilizes a dual-grid approach, where each vertex of the grid has an associated barycentric coordinate. This allows the generation of material boundaries directly from intersections calculated in "barycentric space." The algorithm handles multiple materials and can reconstruct layers and "Y-type" (non-manifold) interfaces. The algorithm does not rely on application-specific knowledge of hydrodynamics or other simulation codes, but solves the problem from a purely mathematical viewpoint.

3. THE TWO-MATERIAL CASE

Consider a grid $\mathcal{S}$ containing n materials, *i.e.*, Each vertex of $\mathcal{S}$ has an associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2 ..., \alpha_n)$. A cell C of $\mathcal{S}$ is called a *two-material cell* if there are two indices i_1 and i_2, such that the associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2 ..., \alpha_n)$. of each vertex of C has the property that $\alpha_i = 0$ for $i \neq i_1, i_2$.

If C is a two-material cell, then without loss of generality, we can assume that each vertex has a barycentric coordinate represented by a two-tuple, $\alpha = (\alpha_1, \alpha_2)$, where $\alpha_1 + \alpha_2 = 1$. Given two points $\mathbf{p}_1$ and $\mathbf{p}_2$ with

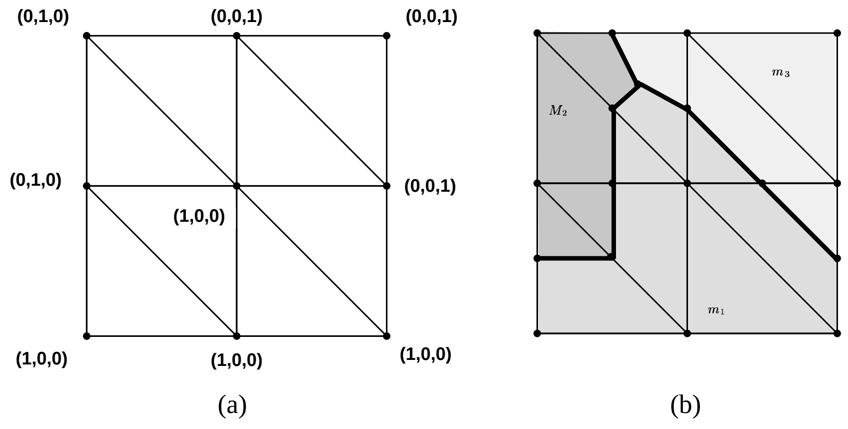


FIGURE 5: Applying the Nielson-Franke algorithm. The test data set (a) represents only "classes" of data, The boundary reconstruction result is shown in (b)

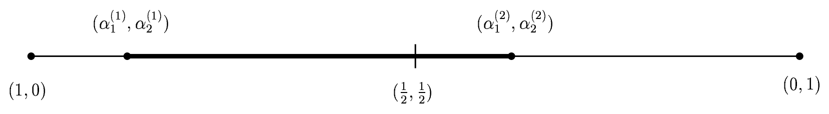

$(\alpha_1^{(1)}, \alpha_2^{(1)})$
$(\alpha_1^{(2)}, \alpha_2^{(2)})$
$(1,0)$
$(\frac{1}{2}, \frac{1}{2})$
$(0,1)$

FIGURE 6: Mapping the barycentric coordinates in the two-material case, results in a line segment, lying on the line having endpoints $(1,0)$, and $(0,1)$. We assume that the material boundary corrresponds to the point $\left(\frac{1}{2}, \frac{1}{2}\right)$

associated barycentric coordinate tuples $\alpha^{(1)}$ and $\alpha^{(2)}$, respectively, the points lie on a line in barycentric space with endpoints $\alpha^{(1)}$ and $\alpha^{(2)}$, as is shown in Figure 6. We assume the material boundary corresponds to the set of points where $\alpha_1 = \alpha_2 = \frac{1}{2}$[2]. There are two cases:

1. The line segment $\overline{\alpha^{(1)}\alpha^{(2)}}$ does not contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the line does not intersect the material boundary.

2. The line segment $\overline{\alpha^{(1)}\alpha^{(2)}}$ does contain the point $(\frac{1}{2}, \frac{1}{2})$. In this case, we assume that the line does intersect the material boundary. We use linear interpolation to calculate a fraction r such that

$$(\frac{1}{2}, \frac{1}{2}) = (1-r)\alpha^{(1)} + r\alpha^{(2)}$$

   and define the point

$$(1-r)\mathbf{p}_1 + r\mathbf{p}_2$$

   as the point on the line $\overline{\mathbf{p}_1\mathbf{p}_2}$ that crosses the material boundary.

[2]This is clearly a heuristic choice, but the most reasonable among all the points of the line.

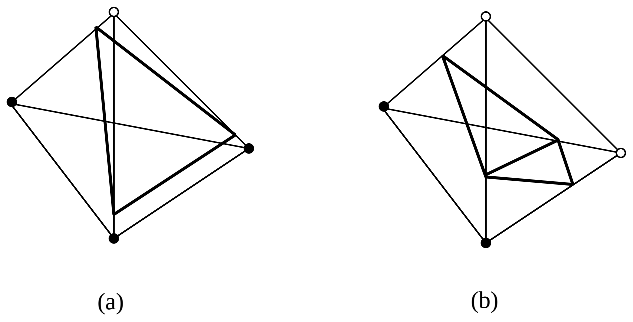


FIGURE 7: Calculating a material interface in the two-material case. A material interface can intersect the tetrahedron in two ways. In case (a), one triangle is produced, in case (b), two triangles are produced.

Suppose T is a triangle contained in a two-dimensional triangular grid with vertices $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, and associated barycentric coordinates $\alpha^{(1)}$, $\alpha^{(2)}$, and $\alpha^{(3)}$, respectively. The triangle can be analyzed with a two-material strategy if the associated barycentric coordinates for the three vertices are such that for all k, $\alpha_i^{(k)}$ and $\alpha_j^{(k)}$ are non-zero for two fixed i and j, and all other entries are zero. In this case, we test the three edges of the triangle for intersection with the material boundary separating materials i and j. Two cases arise:

1. No edge intersects the material boundary. In this case, we assume that the triangle does not intersect the material boundary.

2. Exactly two edges intersect the material boundary. In this case, we calculate the points on the edges where the boundary interface exists and connect the two points with a line.

If T is a tetrahedron contained in a tetrahedral grid, then we can apply the two-material strategy provided the associated barycentric coordinates have at most two non-zero entries, each occurring at one of two materials. In this case, we test the six edges of the tetrahedron and consider the three cases:

1. No edge crosses the material boundary. In this case, we assume that the tetrahedron does not cross the material boundary.

2. Exactly three edges cross the material boundary. In this case, we calculate the points on the edges where the boundary interface intersects and connect these points creating a triangle. This is shown in Figure 7a.

3. Exactly four edges cross the material boundary. In this case, we calculate the points on the edges where the boundary interface intersects and connect these points, creating a quadrilateral. This quadrilateral is split into two triangles. This is shown in Figure 7b.

This is equivalent to the marching-tetrahedra algorithm [5], and therefore, the two-material case is equivalent to an isosurface calculation.

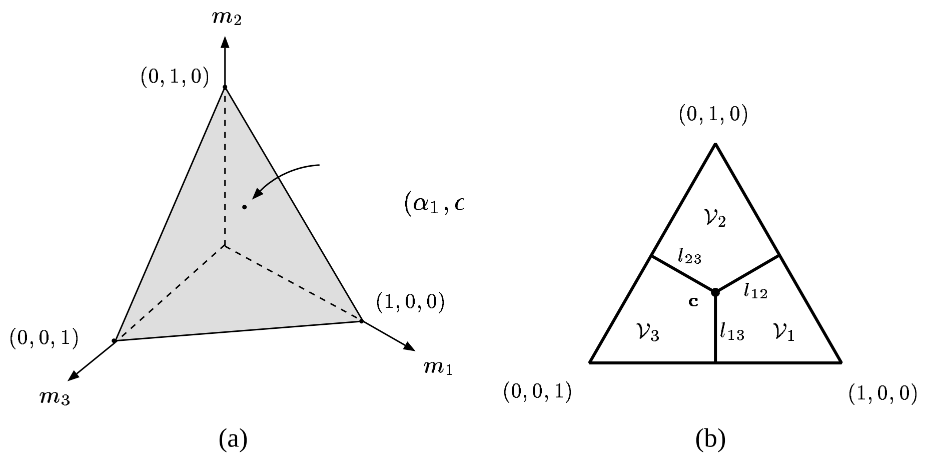


FIGURE 8: The triangle formed by barycentric coordinates of degree three is shown in (a). The partitioned barycentric triangle is shown in (b). The point $\mathbf{c}$ is the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the center of the triangle. The line segments l_{12}, l_{13}, and l_{23} bound the Voronoi cells $\mathcal{V}_j$ in the interior of the triangle.

4. THE THREE-MATERIAL CASE

Consider a grid $\mathcal{S}$ containing n materials, *i.e.*, Each vertex of $\mathcal{S}$ has an associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2..., \alpha_n)$. A cell C of $\mathcal{S}$ is called a *three-material cell* if there are three indices i_1, i_2 and i_3, such that the associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2..., \alpha_n)$. of each vertex of C has the property that $\alpha_i = 0$ for $i \neq i_1, i_2, i_3$.

In the three-material case, it is sufficient to assume that each vertex has an associated 3-tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 + \alpha_2 + \alpha_3 = 1$. Here, α_1 is the fraction of material m_1, α_2 is the fraction of m_2, and α_3 is the fraction of m_3, respectively. The coordinate tuple $(\alpha_1, \alpha_2, \alpha_3)$ lies on the equilateral triangle with vertices $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$, as shown in Figure 8a. The triangle is partitioned into three regions, defined by the *Voronoi cells* $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$, see Figure 8b. The Voronoi cells $\mathcal{V}_j$ are bounded by the edges of the triangle, and the three line segments l_{12}, l_{13}, and l_{23}, where $\alpha_1 = \alpha_2$ and $\alpha_3 \leq \frac{1}{3}$, $\alpha_1 = \alpha_3$ and $\alpha_2 \leq \frac{1}{3}$, or $\alpha_2 = \alpha_3$ and $\alpha_1 \leq \frac{1}{3}$, respectively.

For two-dimensional triangular grids, the associated barycentric coordinates of a triangle T are mapped onto a triangle T' in barycentric space. The intersections of the edges of T' with the edges of the Voronoi cells in the barycentric triangle are used to define material interfaces in T'. These intersections are then mapped back linearly to points in T. There are three cases:

- The triangle T' does not intersect l_{12}, l_{13}, or l_{23}. In this case, it is assumed that no material boundary exists in T.
- The triangle T' intersects at least one of the line segments l_{12}, l_{13}, or l_{23} and the center $\mathbf{c}$ of the barycentric triangle does not lie inside T'. In this case, intersections are calculated on the edges of T, corresponding to

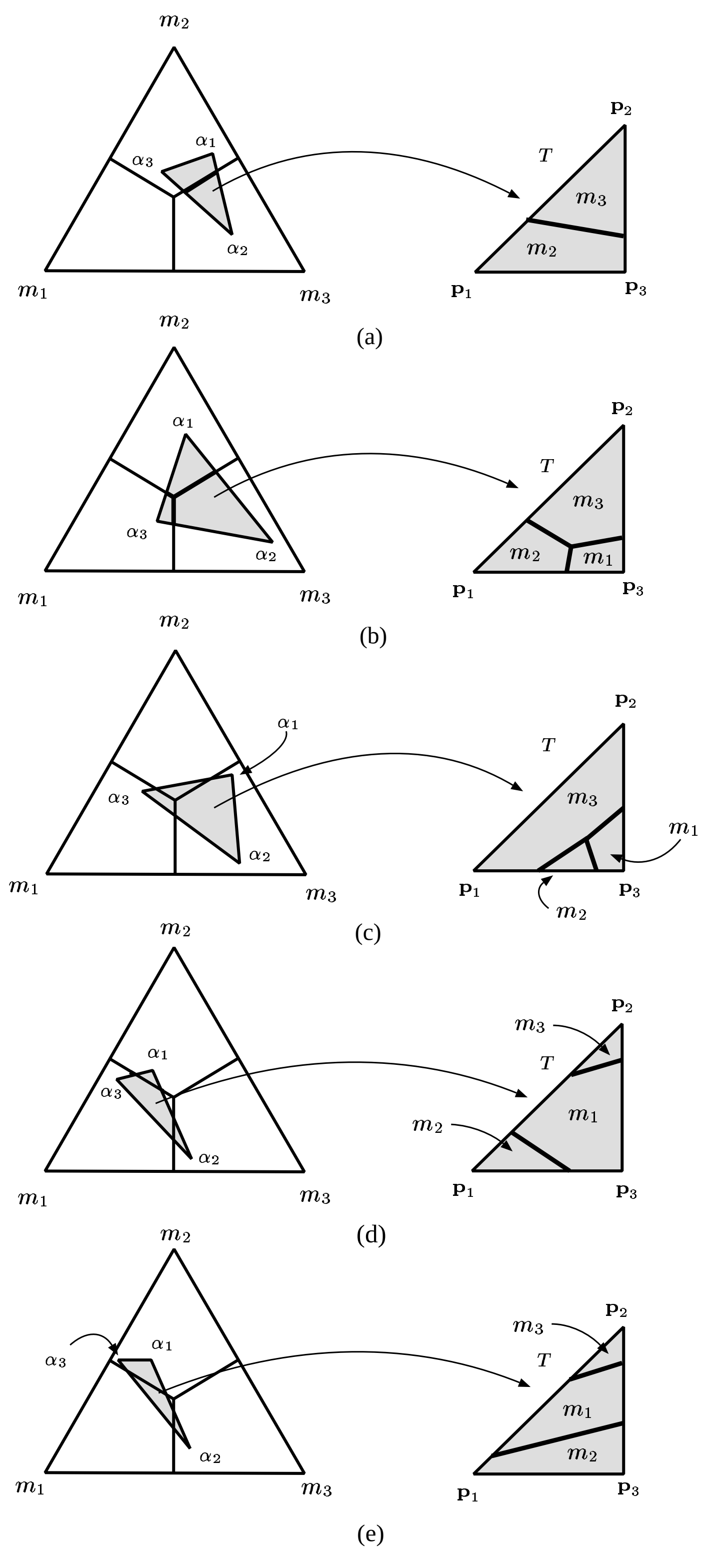


FIGURE 9: Mapping from barycentric space to physical space. The images on the left show the triangle T' in barycentric space, and the images on the right show the material boundary line segments mapped from barycentric space to the original triangle T in physical space.

the intersections of T' with l_{12}, l_{13}, and l_{23}, respectively. (The triangle T' may intersect at most two of these lines.) The material boundary line segments inside T are then defined by the line segments that connect the corresponding edge intersections in T. Figures 9a, 9d, and 9e illustrate these cases.

- The point $\mathbf{c}$ lies inside T'. In this case, three edge intersections are calculated for T, corresponding to the intersections of T' with l_{12}, l_{13}, and l_{23}, respectively. We also determine a point in the interior of T, corresponding to the point $\mathbf{c}$ in T'. The material boundary line segments are defined as the three lines connecting the edge intersections and the face point. Figures 9b and 9c illustrate this case.

If one of the α_i values is zero for each of the three vertices of a triangle, then all points map to an edge of the barycentric triangle. Thus, the situation reduces to the two-material case. If only one material is present at all three vertices, then no intersections are calculated.

For tetrahedral grids, the barycentric values associated with the vertices of a tetrahedron T are used to map the tetrahedron to an image T' of T in barycentric space. Intersections are calculated separately for each face of T', which are then mapped back to T. For the faces of a tetrahedraon, there are three cases to consider:

- No edge of the tetrahedron T' intersects the line segments l_{12}, l_{13}, or l_{23}. In this case, no material boundaries exist in the tetrahedron T.

- The edges of the tetrahedron T' intersect at least one of the line segments l_{12}, l_{13}, or l_{23}, but the point $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, the center of the barycentric triangle, does not lie inside any of the faces of T'. In this case, the intersection line segments for each face of T are calculated and a triangulation is determined from these segments by imitating the triangulation rules for an isosurface extraction algorithm [5]. Figures 10a-d illustrate the possible cases.

- The center point of the barycentric triangle lies inside two faces of T'. In this case, two faces have a single material boundary line segment connecting two edge intersection points, and two faces have three material boundary line segments meeting in the interior of these faces. The intersections are mapped back linearly to the tetrahedron T, using linear interpolation. Using the material boundary line segments for each face, and the line segment connecting the two points in the interior of two faces of T, a valid triangulation of the boundary surface can be determined. Figures 10e-g illustrate the possible cases.

5. THE GENERAL CASE

Consider a grid $\mathcal{S}$ containing n materials, *i.e.,* Each vertex of $\mathcal{S}$ has an associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2 ..., \alpha_n)$. A cell C of $\mathcal{S}$ is called a *k-material cell* if there are k indices $i_1, i_2, ..., i_k$, such that the associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2 ..., \alpha_n)$. of each vertex of C has the property that $\alpha_i = 0$ for $i \neq i_1, ..., i_k$. It is easiest to examine the k-material case, by first looking at the four-material case.

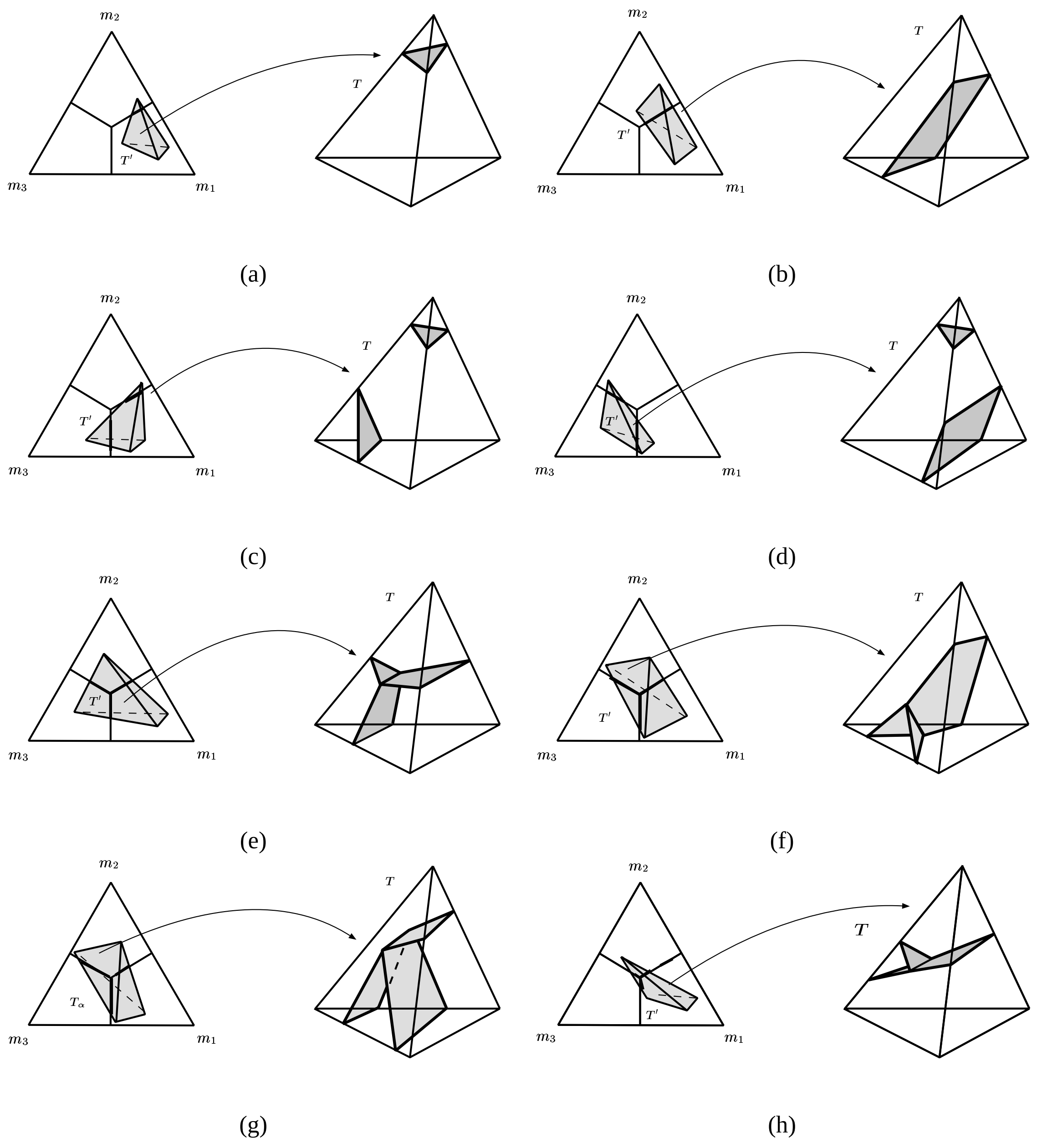


FIGURE 10: Material boundary construction for tetrahedral grids.

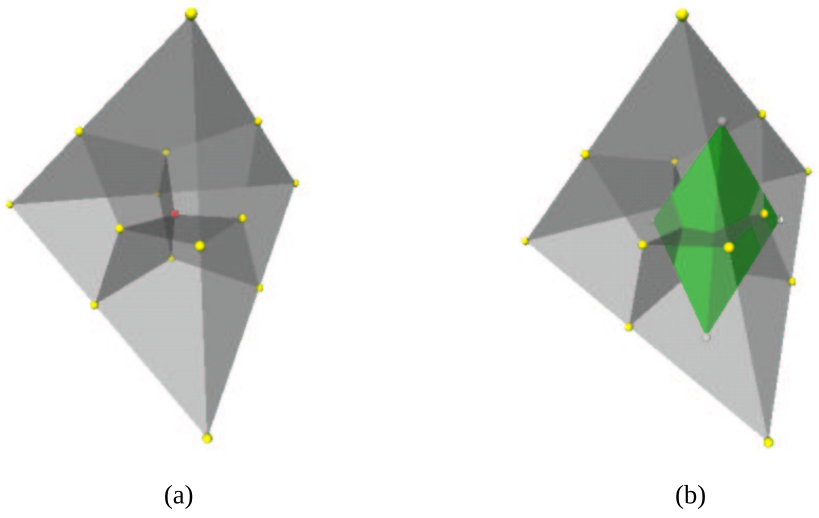


FIGURE 11: Voronoi cell decomposition for the four-material case. The figure illustrates a three-dimensional projection of the barycentric tetrahedron from four-dimensional space. The tetrahedron is segmented into four Voronoi cells in (a). A tetrahedron, mapped from physical space, is shown inside the barycentric tetrahedron in (b).

In the case of four materials, it is sufficient to assume that each vertex has an associated barycentric coordinate given by a four-tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, where $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$, and $\alpha_i \geq 0$. By considering the tetrahedron having vertices $(1, 0, 0, 0)$, $(0, 1, 0, 0)$, $(0, 0, 1, 0)$, and $(0, 0, 0, 1)$ in four-dimensional space, a partition of this tetrahedron can be constructed similarly to the three-material case. Again, the Voronoi cells are used for the decomposition of the barycentric tetrahedron. The boundaries of these cells include parts of the faces of the tetrahedron and six planar pieces, which are defined by $\alpha_1 = \alpha_2$, $\alpha_1 = \alpha_3$, $\alpha_1 = \alpha_4$, $\alpha_2 = \alpha_3$, $\alpha_2 = \alpha_4$, and $\alpha_3 = \alpha_4$. This Voronoi partition is shown in Figure 11a.

For two-dimensional grids, the four-dimensional barycentric coordinates associated with the vertices of a triangle T are mapped into a triangle T' in barycentric space. A clipping algorithm is used to generate the intersections in the triangle T', clipping a triangle against the six planes defining the boundaries of the Voronoi cells of the barycentric tetrahedron. The tetrahedron is stored in a binary space partitioning (BSP) tree, and the clipping algorithm described by Samet [12] is applied. Once the intersections are determined by the clipping algorithm, the material boundary line segments can be determined for the triangle T. For tetrahedral grids, a similar clipping algorithm is used for the image T' of a tetrahedron T. This enables the calculation of the boundary surfaces inside the tetrahedron T', which are then mapped back to the tetrahedron T in physical space.

In the k-material case, a simplex T is mapped to a k-simplex T' in barycentric space. The k-simplex is partitioned into Voronoi cells whose boundaries consist of the faces of the k-simplex and the $\binom{k}{2}$ hyperplanes defined by $\alpha_i = \alpha_j$, where $1 \leq i < j \leq k$. The material boundaries for T' are calculated by using a clipping algorithm and are then mapped back to physical space to form the material boundaries inside T. A BSP algorithm is utilized to perform the clipping.

6. DISCUSSION

The algorithm presented in this paper runs in effectively the same time as does the marching cubes/tetrahedra algorithm. The cells of a grid are traversed, and we calculate, for each cell, a polygonal representation of the material boundaries. Most grid cells in common examples contain only one material, and boundaries do not exist in these cells.

The algorithm can miss material boundaries in tetrahedra. In any isosurface-type algorithm, it is possible for the isosurface to enter a tetrahedron, but only intersect one edge. In this case, the algorithm cannot detect the material boundary from only the vertex information alone. As illustrated in Figure 10a, when a tetrahedron has non-zero fractions of all three materials, it is possible that only two materials are extracted.

In the three-material case, the point $\mathbf{c} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ has been chosen as the "center" of the barycentric triangle. This assumes that there are three distinct sectors in the barycentric triangle, subdividing the triangle in a "Y" fashion, and that a cell of infinitesimally small size contains about one-third of each material in the cell. This is not always the case. For example, consider a "T intersection," where any small cell contains one-half of one material and one-quarter of the other two materials. The segmentation of the barycentric triangle can be adjusted so that the point $\mathbf{c}$ is at an arbitrary location in the triangle, and the edges that determine the intersections can be adjusted appropriately. This can be done by sampling in a larger neighborhood of a specific cell to understand how to weigh the materials about the "Y point." This is a global process: neighboring cells must agree with the change in order to maintain continuity.

In the four-material case, the center of the tetrahedron can also be adjusted. However, this implies that the center vertices on the faces must also be adjusted so that the separating surfaces remain planar, and this then affects the adjacent tetrahedra. In the m-material case, similar considerations also hold when adjusting the center of the m-simplex.

The algorithm presented here is a direct generalization of the Nielson-Franke algorithm [10]. Each vertex of a grid $\mathcal{S}$ has an associated barycentric coordinate $\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$, and by restricting material fractions such that exactly one $\alpha_i = 1$, the case is obtained where each vertex is only associated with one material. In this case, our algorithm produces the same results produced by the Nielson-Franke algorithm.

7. RESULTS

We have generated material interfaces for a variety of data sets. Figure 12 illustrates the material interfaces for a data set consisting of three materials. The boundary of the region containing material 1 has a spherical shape, and the other two material regions are formed as concentric layers around material 1 – forming two material interfaces. The original grid is rectilinear-hexahedral, consisting of $64 \times 64 \times 64$ cells. The dual grid was constructed, and each dual cell was split into six tetrahedra, see Nielson [1], creating 1,572,864 tetrahedra. Approximately 30% of the tetrahedra containing the material boundaries contain two boundary surfaces and require the construction

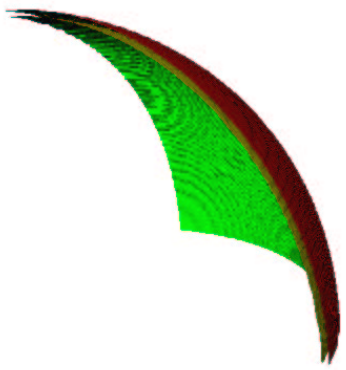


FIGURE 12: Boundary surfaces of three materials defined by two concentric spherical layers.

illustrated in Figure 10c and 10d.

The algorithm generalizes to data sets having several concentric boundary layers. If we have n possible materials per cell, the algorithm can return up to $n-1$ boundaries per cell.

Figure 13 shows the material interfaces for a three-material data set of a simulation of a ball striking a plate consisting of two materials. The original data set is rectilinear-hexahedral and has a resolution of $53 \times 23 \times 23$ cells. Again, the dual grid was created, and each dual cell was split into six tetrahedra, creating 28,037 tetrahedra. Four time-steps are shown.

Figure 14 illustrates the material interfaces for a human brain data set. The original grid is rectilinear-hexahedral containing $256 \times 256 \times 124$ cells. Each cell contains a probability tuple defining the probability that a material is present at the point: gray matter, white matter, or other material. The resulting dual data set contains over eight million tetrahedra.

8. ERROR ANALYSIS

Given a data set and an extracted material interface, we can use the generated interface to approximate material fractions for each cell and compare them to the original fractions. Given an original cell C, and a point $\mathbf{c}$ at the center of the cell, the point $\mathbf{c}$ is a vertex of the dual mesh. There is a set of tetrahedra, generated when the cells of the dual mesh are subdivided, that contain $\mathbf{c}$ as a vertex. Each tetrahedron is partitioned into a set of polyhedra, each polyhedron containing a single material. These polyhedra are clipped against the boundaries of C, and the volumes of the clipped polyhedra are added to the volume fractions for C. Normalizing by the volume of the cell, we obtain a set of volume fractions determined by the extracted material interface. This procedure enables us to calculate the difference between the original volume fractions and volume fractions implied by the extracted

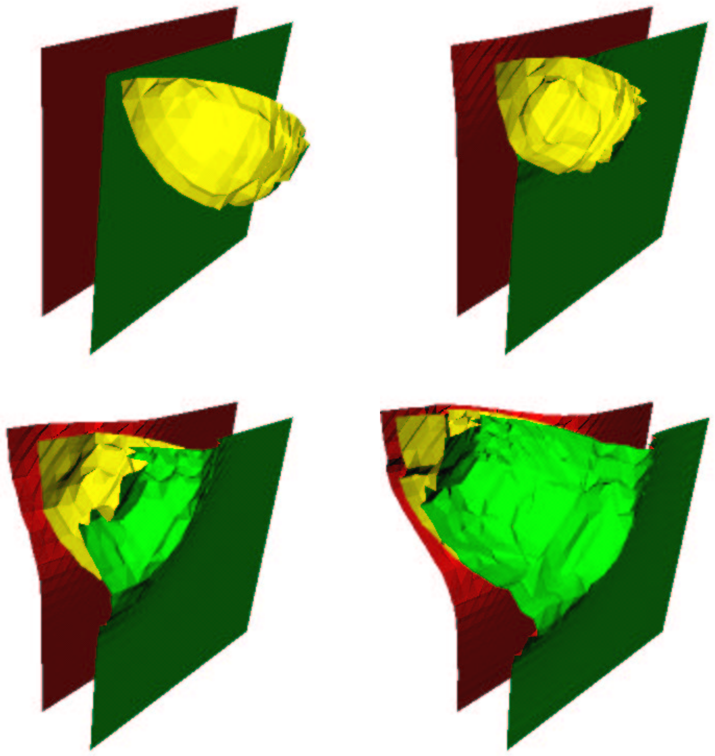


FIGURE 13: Time-dependent simulation of a ball striking a plate consisting of two materials. The sequence shows the boundary surfaces as the ball penetrates the plate.

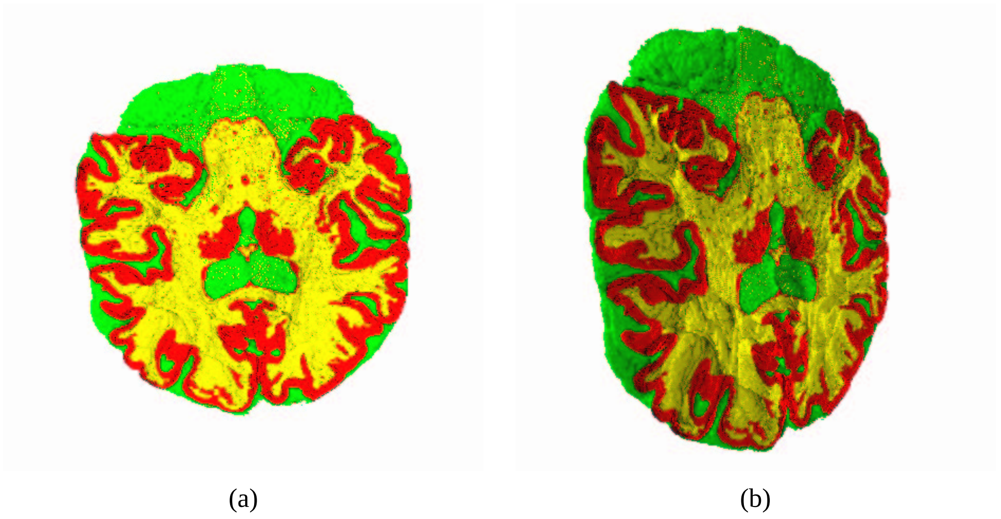
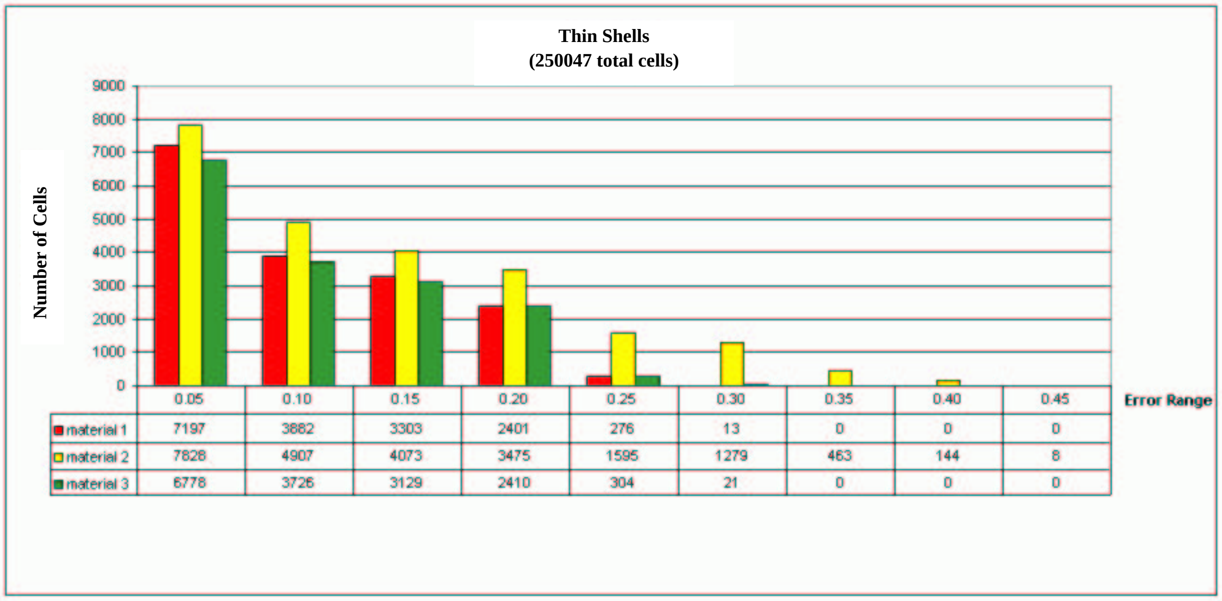


FIGURE 14: Brain data set. Material boundary surfaces are shown in red, green, and yellow. The polygons defining the material boundaries are clipped to show the interior of the data set. Two views of the material boundary surfaces, are shown in (a) and (b).



	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45
🟥 material 1	7197	3882	3303	2401	276	13	0	0	0
🟨 material 2	7828	4907	4073	3475	1595	1279	463	144	8
🟩 material 3	6778	3726	3129	2410	304	21	0	0	0

FIGURE 15: Error analysis for ”thin shells data set” shown in Figure 12.

material interface. It is not accurate on the boundary of the data set since the dual cells do not cover the original cells there.

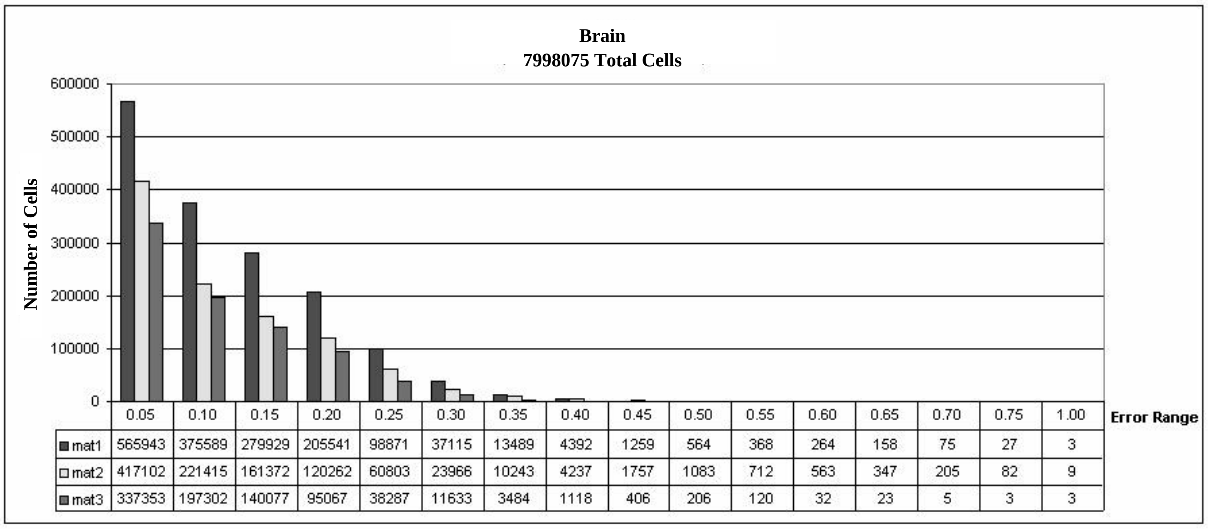
In Figures 15 and Figures 17, errors calculated from the “thin shells” and the “brain” data sets. Errors are reported as numbers of dual cells that fall into a certain error range (Dual cells that contained zero error for a particular material are not reported.) The first error range is the interval (0, 0.05]. Figure 15 shows the errors for the data set shown in Figure 12. There are $250,047$ total dual cells total. The number of zero-error dual cells for materials 1, 2 and 3 are 232975, 226275, and 233679, respectively. Figure 16 provides a comparison of the original and new volume fractions. If we sum the volume fractions for the complete data set and compare it with the calculated fractions, the error is actually quite low. This is evident from Figure 16.

Figure 17 shows the errors for the brain data set shown in Figure 14. The number of zero-error dual cells for materials 1, 2, and 3 are 6414488, 6973917, and 7172956, respectively. Figure 18 provides a comparison of original and new volume fractions.

In general, the approximation faithfully represents the material interface, with little error. We found that in the cells with larger error, most are multiple material cells, where the calculated “Y” point appears in the wrong cell. In these cases, the “Y” point is at most one cell off.

Thin Shells	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.5205060	0.0276901	0.4518040
New volume fractions (summed over dual mesh)	0.5155890	0.0279787	0.4564320
difference:	0.0049170	0.0002886	0.0046280

FIGURE 16: Summing the total fractions over the mesh.



	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	1.00
■ mat1	565943	375589	279929	205541	98871	37115	13489	4392	1259	564	368	264	158	75	27	3
□ mat2	417102	221415	161372	120262	60803	23966	10243	4237	1757	1083	712	563	347	205	82	9
▨ mat3	337353	197302	140077	95067	38287	11633	3484	1118	406	206	120	32	23	5	3	3

FIGURE 17: Error analysis for the brain data set in Figure 14.

Brain	**material 1**	**material 2**	**material 3**
Original volume fractions (summed over original mesh)	0.1177090	0.0840189	0.7982720
New volume fractions (summed over dual mesh)	0.1225630	0.0840548	0.7933830
difference:	0.0048540	0.0000359	0.0048890

FIGURE 18: Summing the total fractions over the mesh.

9. CONCLUSIONS

In this paper, we have presented a new algorithm for material interface construction from data sets containing volume-fraction information. A given grid is transformed to a dual grid, where each vertex has an associated barycentric coordinate tuple that represents the fractions of each material. After subdividing the dual grid into simplices, the material interfaces are constructed by mapping each simplex to barycentric space, calculating the intersections with Voronoi cells in barycentric space. These intersection points are mapped back to physical space and triangulated to form the resulting boundary surface approximation. The algorithm can treat any number of materials per cell, and since it is based on simplicial grids, it can be used for any grid structure.

In the future, we would like to add a “measure-and-adjust” feature to this algorithm. Once an initial boundary surface approximation is calculated, the calculation of (new) volume fractions can be done directly from this boundary surface approximation, as shown in Section 8. It is then possible to adjust material interfaces to minimize volume fraction deviations. It may also be possible to adjust the material interface within each simplex (or higher-level cell), to “optimize” the material interface. If so, it will be possible to preserve volume fractions on a per-simplex (per-cell) basis.

10. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Office of Naval Research under contract N00014-97-1-0222; the Army Research Office under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; and the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159. We thank the members of the Visualization Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis for their support in this work.

References

[1] G. M. Nielson, “Tools for triangulations and tetrahedrizations and constructing functions defined over them,” in *Scientific Visualization: Overviews, Methodologies, and Techniques* (G. M. Nielson, H. Hagen, and H. Müller, eds.), pp. 429–525, IEEE Computer Society Press, Los Alamitos, CA, 1997.

[2] A. Okabe, B. Boots, and K. Sugihara, *Spatial Tesselations — Concepts and Applications of Voronoi Diagrams*. Wiley Publishing, Chichester, England, 1992.

[3] W. E. Lorensen and H. E. Cline, “Marching cubes: a high resolution 3D surface construction algorithm,” in *Computer Graphics (SIGGRAPH ’87 Proceedings)* (M. C. Stone, ed.), vol. 21, pp. 163–170, July 1987.

[4] G. M. Nielson and B. Hamann, “The asymptotic decider: Removing the ambiguity in marching cubes,” in *Proceedings of IEEE Visualization ’91*, pp. 83–91, IEEE Computer Society Press, Los Alamitos, CA, 1991.

[5] Y. Zhou, B. Chen, and A. Kaufman, “Multiresolution tetrahedral framework for visualizing regular volume data,” in *IEEE Visualization ’97* (R. Yagel and H. Hagen, eds.), pp. 135–142, IEEE, IEEE Computer Society Press, Los Alamitos, CA, Nov. 1997.

[6] W. F. Noh and P. Woodward, “SLIC (Simple line interface calculation),” in *Lecture Notes in Physics* (A. I. van der Vooren and P. J. Zandbergen, eds.), pp. 330–340, Springer-Verlag, 1976.

[7] D. L. Youngs, “Time-dependent multi-matreial flow with large fluid distortion,” in *Numerical Methods for Fluid Dynamics* (K. W. Morton and J. J. Baines, eds.), pp. 273–285, Academic Press, 1982.

[8] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, and S. Zaleski, “Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows,” *Journal of Computational Physics*, vol. 152, pp. 423–456, 1999.

[9] J. E. Pilliod and E. G. Puckett, “Second-order accurate volume-of-fluid algorithms for tracking material interfaces,” technical report, Lawrence Berkeley National Laboratory, 2000.

[10] G. M. Nielson and R. Franke, “Computing the separating surface for segmented data,” in *Proceedings of IEEE Visualization ’97* (R. Yagel and H. Hagen, eds.), pp. 229–234, IEEE Computer Society Press, Los Alamitos, CA, Oct. 1997.

[11] K. Bonnell, D. Schikore, M. Duchaineau, B. Hamann, and K. I. Joy, “Constructing material interfaces from data sets with volume-fraction information,” in *Proceedings of IEEE Visualization 2000* (T. Ertl, B. Hamann, and A. Varshney, eds.), pp. 367–372, IEEE Computer Society Press, Los Alamitos, CA, Oct. 2000.

[12] H. Samet, *The Design and Analysis of Spatial Data Structures*. Series in Computer Science, Addison-Wesley, Reading, Massachusetts, 1990.

[13] K. Bonnell, “On material boundary surfaces,” M.S. thesis, Department of Computer Science, University of California, Davis, June 2000.